APPENDIX: DEMONSTRATION

In the first phase of the demonstration, we will show how the Pac-Man DSL is defined in the language workbench of the GEMOC Studio. We will first show how the non-reactive language is defined, and then show how it is made reactive by annotating execution rules from the operational semantics as event handlers. Then, we will give an overview of the artifacts generated from these annotation and the definition of the DSL, that is the behavioral interface for the Pac-Man DSL.

In the second phase of the demonstration, we will show how a game can be defined using an editor we implemented on top of the abstract syntax of the Pac-Man DSL. We will then demonstrate the result by playing a Pac-Man game.

We will then go back to the language workbench, and annotate the *modifySpeed* execution rule as an event. After regenerating the behavioral language interface, we will bind the new event to a key and then launch another game, in which we will be able to send the new event to cheat by increasing the speed of the pacman.

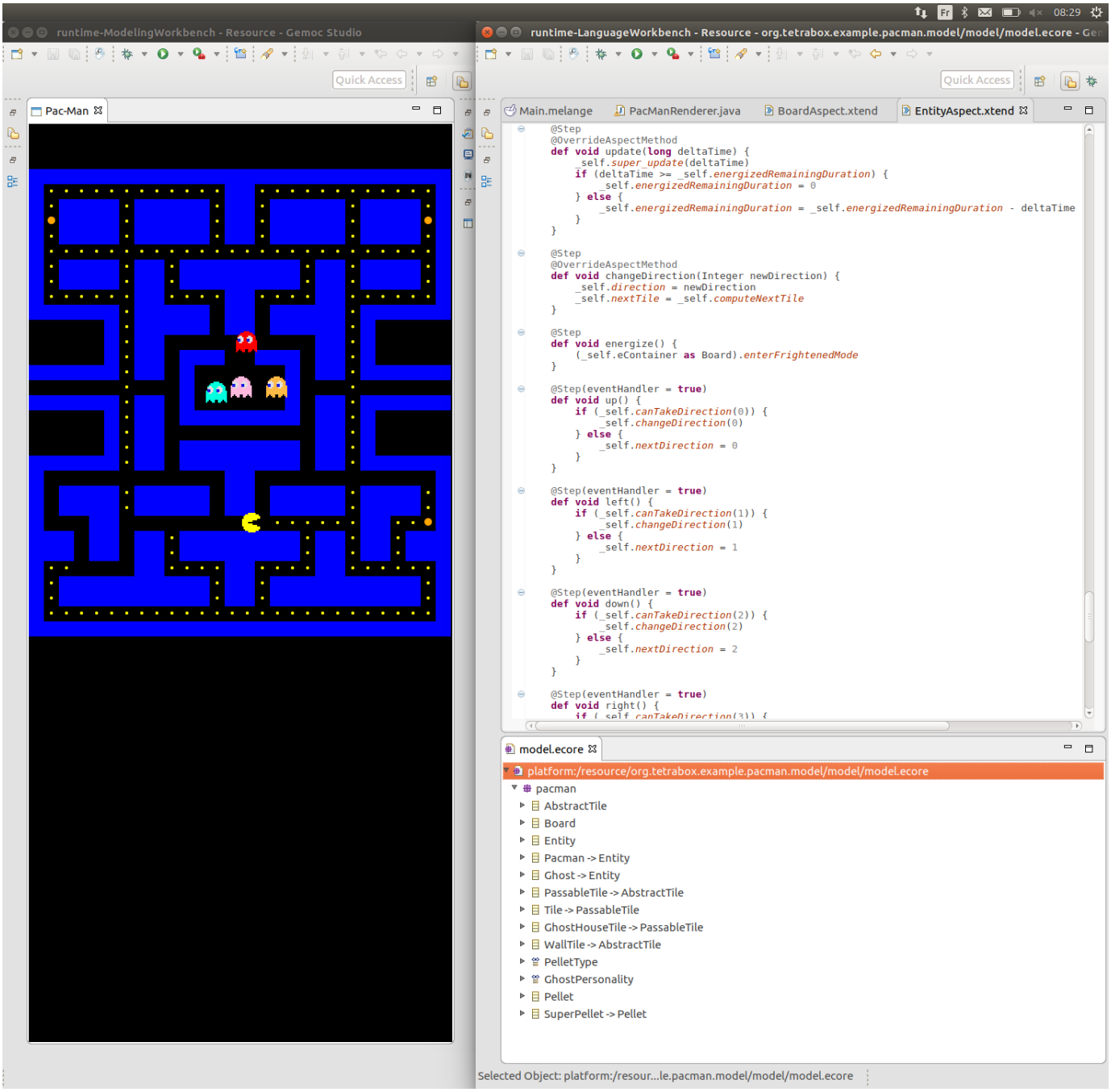

Fig. 4: Overview of the language workbench and of a Pac-Man game being played.